DESI, ITESO A.C.

Memory Allocation Specification

Embedded Systems

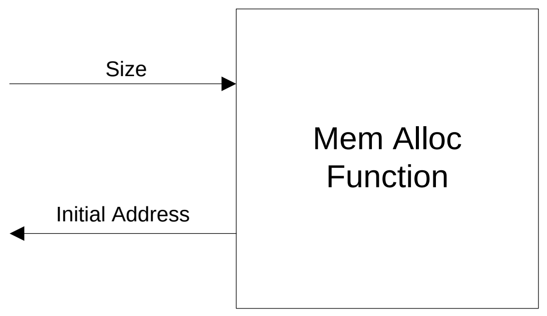
Francisco Martinez Chavez

3-Sep-18

Author	Francisco Martínez Chávez
Date	3-Sep-18

1. TABLE OF CONTENTS

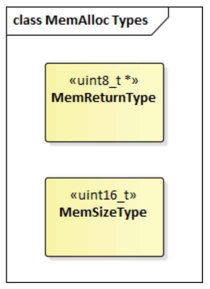
2. FUNCTIONAL SPECIFICATION

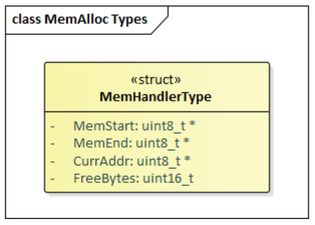


1 MEMORY ALLOCATION DATA FLOW DIAGRAM

2.1. TYPE DEFINITIONS

Types used by the memory handler shall be declared as follows:

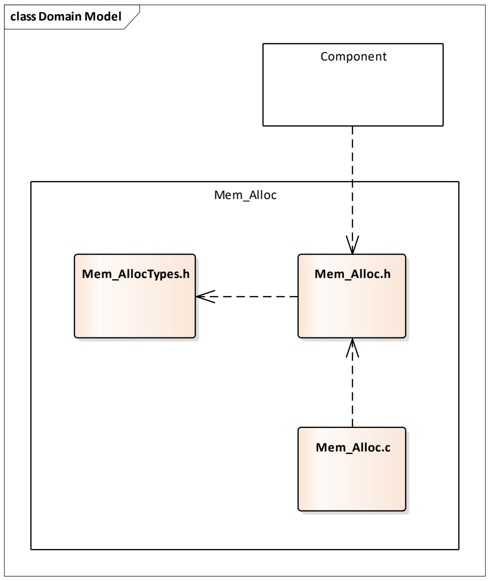




Those type definitions shall be used internally by the MemAlloc Handler.

2.2. File Structure

The image below the file dependencies where *A → B* indicates A includes B.



The table below shows a brief description of each file.

File	Description
Mem_AllocTypes.h	Contains all the internal data types definitions use by the memory allocation handler Module
Mem_Alloc.h	Contains all the interfaces provided to the user component modules
Mem_Alloc.c	Contains the main functionality of the memory allocation handler

2.3. HEAP MEMORY AREA

2.3.1. LINKER CONFIGURATION

Memory Allocation area name shall be "heap_memalloc". The heap_memalloc **space** shall be allocated at the RAM location 0x20400000. The total size of this space shall be 64KB.

The heap_memalloc space and corresponding section references shall be provided from the Linker Configuration File (sam_flash.ld).

Additional heap_memsize configuration in the Linker Configuration File shall be provided. The heap_memsize is the actual heap space to be used in the project.

Initial heap_memsize configuration shall be of 4KB.

```
/* ********************************************
 *
 *  File: samv71q21_flash.ld
 *  Heap Space and size configuration example
 *
 ********************************************/

/* Memory Spaces Definitions */
MEMORY
{
    rom (rx)                  : ORIGIN = 0x00400000, LENGTH = 0x00200000
    heap_memalloc(rwx) : ORIGIN = 0x20400000, LENGTH = 0x00010000
    ram (rwx)                : ORIGIN = 0x20410000, LENGTH = 0x00050000
    sdram(rwx)               : ORIGIN = 0x70000000, LENGTH = 0x00200000
}

/* The stack size used by the application. NOTE: you need to adjust according to your application. */
STACK_SIZE = DEFINED(STACK_SIZE) ? STACK_SIZE : 0x2000;

/* The heapsize used by the application. NOTE: you need to adjust according to your application. */
HEAP_SIZE = DEFINED(HEAP_SIZE) ? HEAP_SIZE : 0x1000;

heap_memsize = DEFINED(heap_memsize) ? heap_memsize : 0x1000;

INCLUDE ..\..\..\toolset\gcc\sam_flash.ld
INCLUDE ..\..\..\toolset\gcc\sam_sdram.ld
```

The startup code shall be updated so that the heap memory is set to the value of zero.

The following labels to support this functionality shall be named as follows:

- _heap_mem_start
- _heap_mem_end

Hint! The above labels are created in sam_flash.ld file.

The data struct MemHandlerType elements (memory start address *MemStart*, memory end address *MemEnd*, memory current address *CurrAddr* and available memory bytes indicator *FreeBytes*) shall be statically initialized in the Mem_Alloc.c file.

```
/* ********************************************
 *
 *  File: Mem_Alloc.c
 *  MemControl data initialization example
 *
 ********************************************/
…
MemHandlerType MemControl =
{
      .MemStart = (uint8_t *) &_heap_mem_start,                                    /* Sets the start of the heap memory */
      .MemEnd = (uint8_t *) &_heap_mem_end,                                        /* Sets the end of the heap memory */
      .CurrAddr = (uint8_t *) &_heap_mem_start,                                    /* Initialize the current start address */
      .FreeBytes = (uint8_t *) &_heap_mem_end - (uint8_t *) &_heap_mem_start;  /* Sets the size of the heap memory */
};
…
```

2.4. Memory Handler

2.4.1. Memory Allocation

Memory allocation handler is shown below.

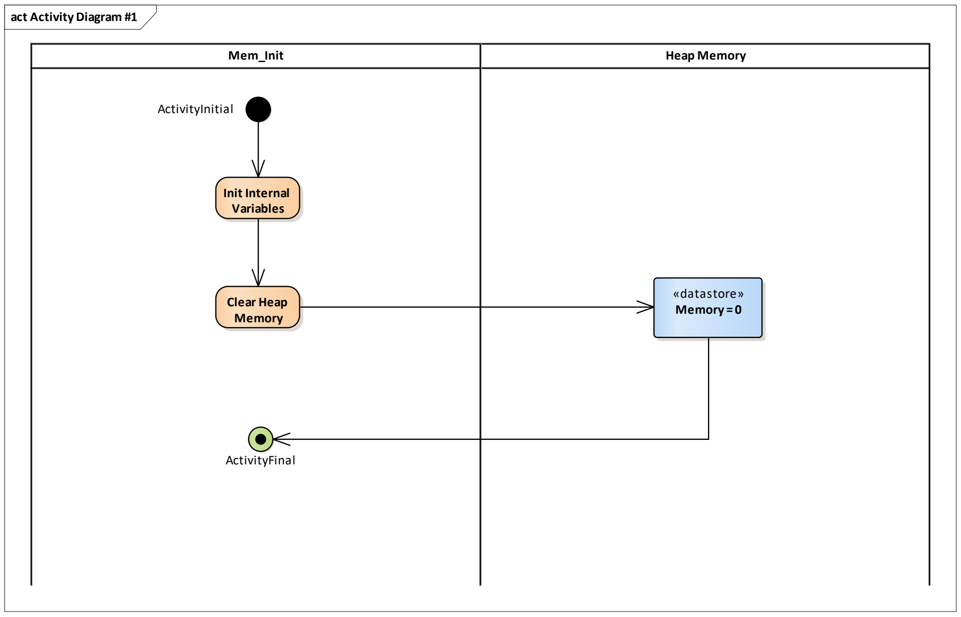
Service Name	Mem_Alloc	
Syntax	Mem_ReturnType Mem_Alloc (MemSizeType Size)	
Sync/Async	Syncronous	
Reentrancy	Non-Reentrant	
Param (in)	MemSizeType Size	Size in Bytes to be allocated
Param (out)	None	
Return Value	MemReturnType	Initial address of the new allocated memory space
Description	Allocates and returns the initial address of the memory currently being allocated	

- Memory Allocation shall be invoked when memory allocation is requested by the project specific module initialization.
- Mem_Alloc shall return the initial address of the new allocated memory space.
- Current Address *CurrAddr* shall be updated according to the requested size.
- After allocating a new area, Mem_Alloc shall assure the current address is aligned with 32bit address.
- The available memory in the heap *FreeBytes* shall be updated accordingly.
- Mem_Alloc shall return a NULL pointer and the requested memory allocation shall not be handled if the size exceeds the available memory in the heap.

2.5. Sequence and Activity Diagrams

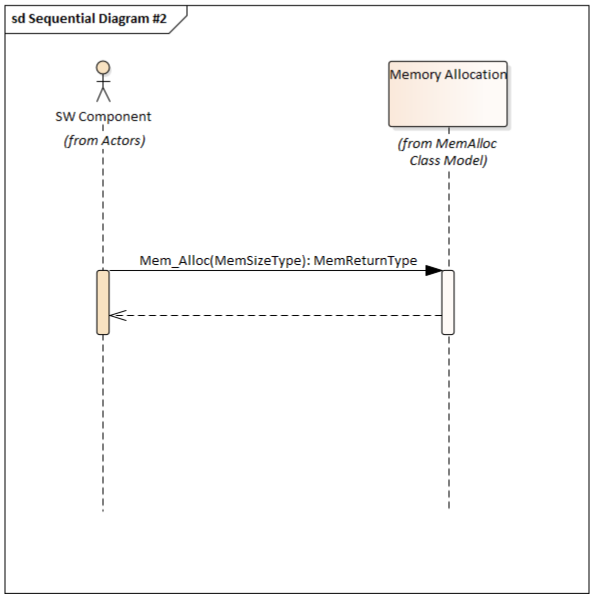
2.5.1. Memory Initialization Activity

The basic steps to initialize the heap memory supported by the startup code:



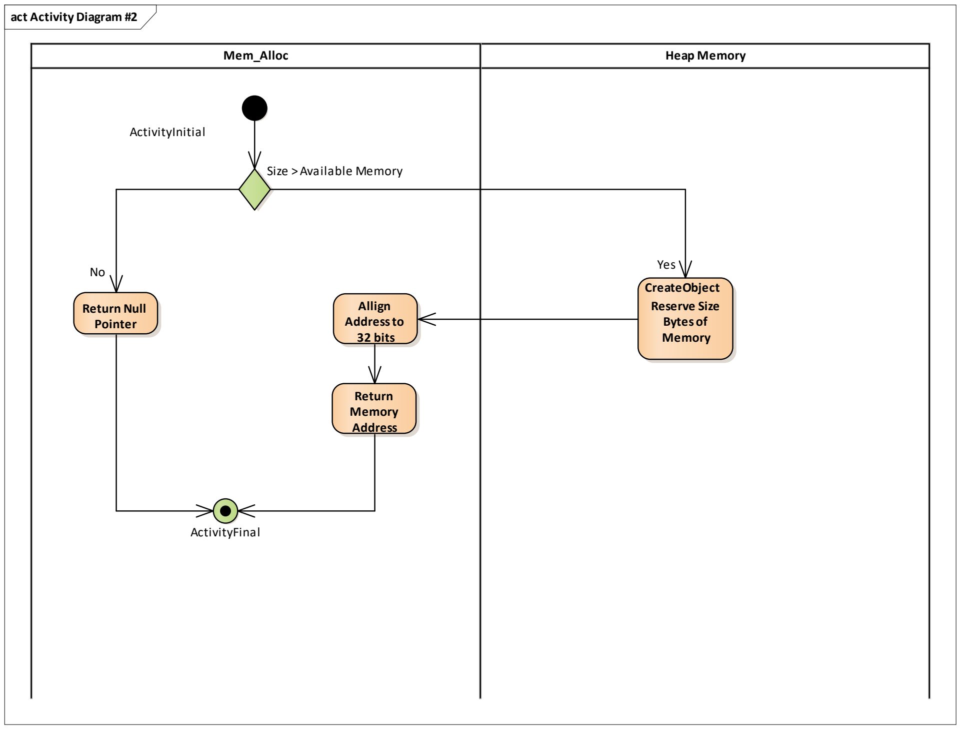
2.5.2. Memory Allocation Sequence

The basic memory allocation sequence is shown below:



2.5.3. Memory Allocation Activity

The basic steps to allocate memory through the invocation of Mem_Alloc are shown below:


act Activity Diagram #2
Mem_Alloc
Heap Memory
ActivityInitial
Size > Available Memory
No
Yes
Return Null Pointer
Allign Address to 32 bits
CreateObject Reserve Size Bytes of Memory
Return Memory Address
ActivityFinal